



EUROPEAN PATENT APPLICATION

⑤¹ Int. Cl.⁵: **G06F 3/06**

②② Date of filing : 09.03.94

**(72) Inventor : Niijima, Hideto
2-42-9, Minami-otsuka,
Toshima-ku
Tokyo (JP)
Inventor : Toyooka, Takashi
4-19-47, Minami-Ikuta,
Tama-ku
Kawasaki-shi, Kanagawa-ken (JP)**

⑦4 Representative : **Davies, Simon Robert**
I B M
UK Intellectual Property Department
Hursley Park
Winchester, Hampshire SO21 2JN (GB)

⑦4 Representative : **Davies, Simon Robert**
I B M
UK Intellectual Property Department
Hursley Park
Winchester, Hampshire SO21 2JN (GB)

⑦① Applicant : International Business Machines Corporation
Old Orchard Road
Armonk, N.Y. 10504 (US)

(54) Nonvolatile memory.

(57) A nonvolatile memory (20) with flash erase capability comprises a plurality of sectors, each of the sectors holding the attribute information for identification. A cluster information sector is in principle placed in the top of a cluster to which it belongs. A data sector is placed in a data area which is the region other than the top of the cluster. A controller (30) connected to the memory creates a cluster information copy sector when erasing a cluster, and reconstructs cluster management information from the cluster information copy sector when initializing a cluster, thereby forming a cluster information sector.

This improves the endurance against failure such as power failure in a solid state file apparatus using a nonvolatile memory with flash erase capability.

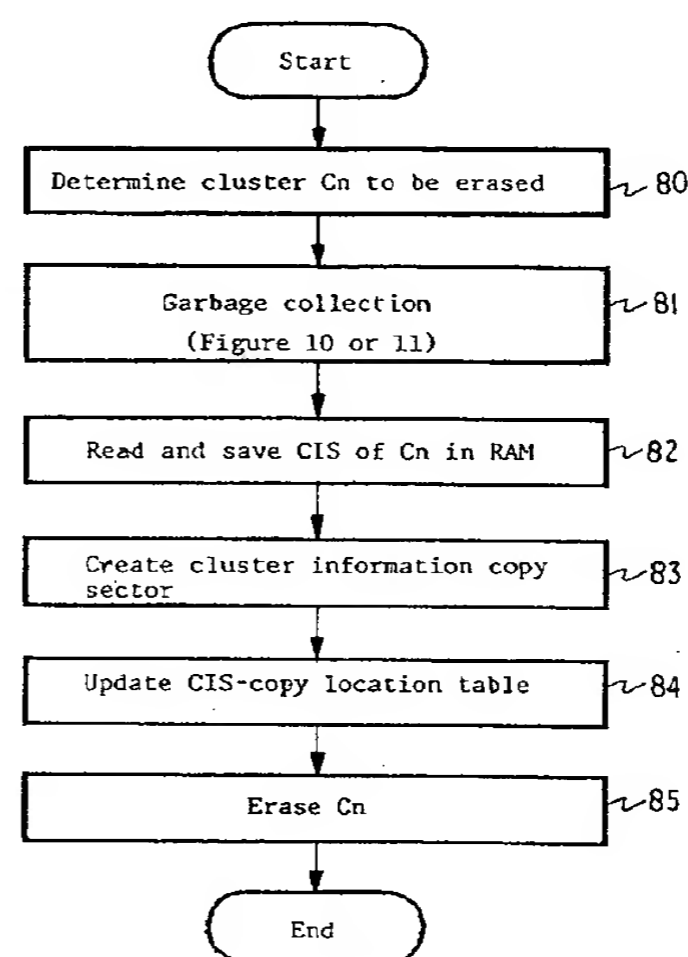


FIG. 8

The present invention relates to a nonvolatile memory with flash erase capability such as a flash EEPROM (hereinafter referred to as a flash memory) and a solid state file apparatus using the same, and more particularly to save, recover, and update the control information such as erase counts of a cluster.

A flash memory has been developed as a memory used for a solid state file apparatus. Its memory element is composed of one transistor as a DRAM so that it can be packaged at high density, and it is expected to have a bit cost equivalent to, or less than that of a DRAM (low cost, large capacity), depending on the future market. The memory element is nonvolatile and does not require a battery backup. Erasure is generally performed for each chip for each smaller block. The outline of such a flash memory is introduced by Richard D. Pashley et al. in "Flash Memories: the best of two worlds," IEEE SPECTRUM, December 1989, pp. 30 - 33. As far as performance is concerned, the block erase type is superior to the chip erase type.

When the flash memory of block erase type is used for a solid state file (SSF), it is convenient to memory management if the size of a block is made equal to a sector, which is a unit of access in the magnetic disk apparatus. European Patent Application 392895, for example, discloses a flash EEPROM system of the sector erase type. The system makes it possible to simultaneously erase any plural sectors by providing a latch for each sector, which is a unit of erasure, and setting a latch corresponding to a sector to be erased. Also known is a flash memory whose unit of erasure is a block having a size equivalent to a plurality of sectors (e. g. 4K bytes). Furthermore, it is possible to make a cluster including a plurality of blocks, each of which is a physical erase unit, a logical erase unit.

However, the flash memory has limitations which SRAMs and DRAMs do not have. First, the programming of memory bits is a one-way process and change is allowed only from 0 to 1 or from 1 to 0. Therefore, when new data is to be written to a memory location which has already been written, writing should be performed after a block including that memory location is erased to an all 0 or all 1 state. It usually takes from several tens of milliseconds to several seconds for erasure and writing. Furthermore, the flash memory is deteriorated by erasure and writing and reaches a use limit, at present, after several tens of thousands to several hundreds of thousands of erasures and writings.

To cope with such problems, it is necessary to save the erase counts for each block so as not to exceed the upper limit of the erase counts. In addition, as disclosed in Patent Appln. No. 3-197318, it is necessary to provide such measures that bad sectors in a cluster, the locations of which have been previously stored, are skipped when physical sectors are dynam-

ically assigned to logical sectors. Although there arises such a situation where control information describing the state of flash memory should be held in the flash memory itself, because of the characteristics of the flash memory, such control information is erased at the same time data is erased. Usually, such control information is copied in a RAM area of the controller for SSF prior to its erasure, and written back into the flash memory after the blocks are erased. However, when such an approach is employed, if a system failure such as power failure should happen during erasure of clusters, the control information on the RAM is lost and the controller cannot recover it.

In addition, block failure may occur in the flash memory because of its construction. Such a block failure may prevent the system from reading the control information. If the control information is lost due to such a failure mode, not only does the system lose the data in that block, but it may also become impossible to control the integrity of the entire SSF. To date, no effective method has been proposed for data protection management against a block failure.

The control information includes data very critical for managing the system such as the erase count of a block which is never recovered once lost so that loss of it causes a severe problem in controlling the system. There has been no measures which can cope with a plurality of failure modes in protecting data which should be held at all times.

In addition, when a physical sector for holding the control information is determined at a fixed location, if a failure should occur in that sector, the entire block or cluster controlled by the control information becomes unusable. That is, there arises such inefficiency that a failure in a critical sector causes an area several tens times greater to fail logically although there is no physical failure. To prevent this, there is a method to provide a replacement sector for a critical sector. In this case, however, it is necessary to store in the system where the replacement sector is. In general, the information about the replacement sector is required to be stored on a flash memory. In consideration of the characteristics of flash memories, this information should be stored in a suitable sector belonging to a cluster containing the sector to be alternated. Under such conditions, the information on the location of replacement sector in view of its nature may be stored in the flash memory as part of control information. However, such an approach causes a contradiction in that to locate the control information requires the information to refer to itself (for the location of the replacement sector). To date, no effective measures have been proposed for such a situation.

In addition, it becomes necessary to update the control information (bad sector map) due to the occurrence of a new bad sector or the like. However, because, in general, the flash memory cannot be overwritten, the update should wait until the control infor-

mation is written back after the completion of erasure of that cluster. The problem of how such update transaction should be held until the completion of update processing is also yet unsolved.

Viewed from one aspect the present invention provides a nonvolatile memory having flash erase capability, comprising a plurality of sectors, each of the sectors storing attribute information for identification.

Viewed from another aspect the present invention provides a solid state file apparatus comprising: a controller; and a nonvolatile memory with flash erase capability comprising a plurality of clusters, each of said clusters comprising a plurality of sectors, said clusters being erasable cluster-by-cluster basis, each cluster being provided with a cluster information sector for storing management information for said cluster, said cluster information sector holding the attribute information for cluster information sector, and sectors for user data holding the attribute information for data sector.

According to the present invention, endurance against failure such as power failure can be improved for a solid state file apparatus using a nonvolatile memory with flash erase capability.

In order that the invention may be fully understood preferred embodiments thereof will now be described, by way of example only, with reference to the accompanying drawings in which:

Figure 1 shows a block diagram illustrating an example of a computer system which uses a flash memory according to the present invention as a solid state file apparatus;

Figure 2 shows a block diagram illustrating a schematic configuration of the solid state file apparatus;

Figure 3 shows a format of a cluster information sector in the flash memory of the first embodiment;

Figure 4 shows a format of a data sector in the flash memory;

Figure 5 shows a format of a cluster information copy sector in the flash memory;

Figure 6 shows a format of a bad sector information sector in the flash memory;

Figure 7 shows a flowchart illustrating the operation of the SSF when a bad sector is detected;

Figure 8 shows a flowchart illustrating the operation of the SSF when a sector is erased;

Figure 9 shows a flowchart illustrating the operation of the SSF when a sector is initialized;

Figure 10 shows a flowchart illustrating an example of garbage collection;

Figure 11 shows a flowchart illustrating another example of garbage collection;

Figure 12 shows a flowchart illustrating the operation of the SSF when an address translation table is reconstructed;

Figure 13 shows a flowchart illustrating the oper-

ation of the SSF when an address translation table is reconstructed;

Figure 14 shows a flowchart illustrating reading of the top sector of a cluster of the SSF in the first embodiment and operations related thereto;

Figure 15 shows a relationship between a block and pages in the second embodiment;

Figure 16 shows a relationship between clusters, blocks, sectors and pages in the second embodiment;

Figure 17 shows a format of a cluster information sector in a flash memory in the second embodiment;

Figure 18 shows a relationship between a replacement-CIS pointer sector and a replacement-CIS;

Figure 19 shows a format of a replacement-CIS pointer sector;

Figure 20 shows a flowchart illustrating reading of the top sector of a cluster of the SSF in the second embodiment and operations related thereto; and

Figure 21 shows a flowchart illustrating reading of the top sector of a cluster of the SSF in the second embodiment and operations related thereto.

Figure 1 shows an example of a computer system in which a solid state file apparatus according to the present invention is incorporated. A CPU 10 communicates, through a system bus 13, with a main storage 15, a bus controller 16, and an optional math coprocessor 14. Communications between the CPU 10 and the peripheral equipment are performed through a bus controller 16. To this end, the bus controller 16 is connected, through a family bus 18, to the peripheral equipment. A solid state file apparatus (SSF) 20 made of the flash memory according to the present invention, a communication device 21, a floppy disk drive (FDD) 22, an optical file unit 23, and a display device 24 are connected to the family device 18 as the peripheral equipment. Of course, other peripheral equipment may also be connected. An example of such a computer system is the IBM PS/2 computer system.

A direct memory access controller (DMAC) 12 is provided to enable memory access by all or some selected peripheral equipment. To this end, at least a portion of the family bus 18 is branched to the DMAC 12. Each peripheral equipment which is allowed for DMA is provided with an arbitration circuit, though not shown in the drawing, and is assigned an arbitration level (priority). In association with the DMAC 12, a central arbitration control circuit 11 provided which arbitrates among a plurality of peripheral equipment simultaneously requesting the DMA and informs the DMA 12 which peripheral equipment is granted the DMA. Details of the DMA control by the DMAC 12 and the central arbitration control circuit 11 are described in U. S. Patent 4,901,234.

The CPU 10 uses the SSF 20 as a hard disk drive. Therefore, when the SSF 20 is accessed, a relative block address (RBA) comprising a head number, a cylinder number, and sector number, is sent to the SSF 20. The SSF 20 performs dynamic sector allocation. Therefore, the relationship between the RBA provided by the CPU 10 and an address (physical address) of a block of the SSF 20, which is actually accessed, is not fixed and varies each time writing is performed. Then, an address translation table is provided for indicating the relationship. That is, the RBA from the CPU 10 is a logical address. A corresponding physical address is written in the address translation table at an entry pointed by a logical address.

Figure 2 shows a schematic configuration of the SSF 20. The SSF 20 comprises a controller 30 connected to the family bus 18, and an internal bus 31 connected to a random access memory (RAM) 32, a bus control element 33, and a flash memory 34. Although not shown, the controller 30 includes a microprocessor and a ROM storing a program controlling it. The functions such as erasure, which are explained by referring flowcharts in Figure 7 and thereafter, are implemented by executing the program in the ROM with the microprocessor.

The RAM 32 includes an area 35 for storing the address translation table, and a buffer area 36. In addition, the RAM 32 includes a table 38 for storing the location of a cluster information copy sector (hereinafter called the "CIS-copy location table"), and a table 39 for storing the location of a bad information sector (hereinafter called the "bad sector information location table"), both of which are described later. The bus control element 33 has the well-known receiver/driver configuration for interconnecting the internal bus 31 and a memory bus 37 connected to the flash memory 34.

Two embodiments with different configurations of sectors and clusters will be described in the following. Both of them are assumed that the sector size specified by the CPU 10 is 512 bytes, while the size of a physical sector, which is the minimum access unit of the CPU 10 to the SSF 20, is 512 bytes + α . In the first embodiment, sectors (physical sector) and clusters of the SSF 20 are managed as follows.

1) A logical set is created for which actual erasure is performed, and which is called a cluster. The cluster consists of one or more blocks, each of which is a physical erase unit. In this embodiment, eight sectors constitute one block, and eight blocks constitute one cluster. The relationship between the block and the cluster is established by making the upper address of the block an identifier for a cluster, or by creating a table.

2) In this embodiment, each of the physical sectors has the following attribute, and is identified by an attribute flag.

First, a sector positioned at the top physical ad-

dress of each cluster is assigned to the cluster information sector (CIS) to store management information inherent to the cluster such as the cluster erase counts and the bad sector map in the cluster (see Figure 3). In addition, a redundancy area (α bytes) includes an area in which an attribute flag is written and to which a "cluster information" flag indicating it as the cluster information sector is set. A parity code or the like is added to each of these information units enabling it to detect occurrence of an error.

The parity for each information is preferably to be a code allowing error correction. Here, it is sufficient to be a simple even-odd parity or CRC parity. ECC parity is calculated for entire management information including all of these parities (including the attribute flag), and the result is written in the redundancy area in the cluster information sector.

Japanese Pat. Appln. No. 5-35228 discloses an invention wherein a sequence number is previously provided for each cluster in such a manner that no two clusters have the same sequence number, when erasing and then initializing a given cluster, a value larger than the current maximum sequence number is written into the given cluster as a new sequence number. When clusters are managed by using such sequence number, a sequence number is written in the cluster information sector as management information. The cluster management information does not occupy the entire area of 512 bytes.

The top sector of a cluster is a vital sector in which the cluster management information is placed. However, it may suffer from a failure such as word line failure as in the other sectors. If the top sector is bad, it is arranged so as to use as the cluster information sector a sector preceding other sectors with any other attributes.

Because sectors other than the top sector of a cluster are areas in which user data is written, they are called user areas. Figure 4 shows a format of sector holding user data for each cluster (hereinafter called the "data sector"). As shown in the figure, the data sector includes areas for holding an attribute flag and an error correction code (ECC) in addition to the data area of 512 bytes for holding user data. Set in the attribute flag is a "data" flag indicating it to be a data sector. In this embodiment, dynamic sector allocation is performed so that a reverse pointer pointing an entry of the address translation table is written as part of the attribute flag.

The data area may include sectors of a kind other than the data sector. Figure 5 shows a format of a sector for holding a copy of cluster information sector (hereinafter called the "cluster information copy sector" or "CIS copy sector"). This sector is one for holding a copy of management information for a cluster other than the cluster to which it belongs. As shown in the figure, a cluster number and a copy of management information for the cluster with such number are

written in the 512-byte data area, and a "cluster information copy" attribute flag indicating a cluster information sector and an ECC are written in the α -byte redundancy area.

Figure 6 shows a format of a bad information sector. This sector is a one for holding numbers of bad data sectors found in a cluster to which it belongs and other sectors. As shown in the figure, cluster numbers, bad sector numbers in the cluster with that number, and erase counts at the moment when a bad sector is detected are written in the 512-byte data sector, and a "bad sector information" attribute flag and an ECC are written into the α -byte redundancy area.

It should be noted that the formats shown in Figures 3 - 6 are models. For example, the attribute flag may be positioned in an area which is first accessed in the entire $512 + \alpha$ bytes (top of a word line).

3) Now, the operation of the SSF will be explained by referring to Figure 7 when a bad data sector is found as a result of verification accompanying writing into a sector.

If a verification error occurs even after writing in a data area several times, the controller temporarily save the cluster number, the erase count, and the sector numbers of the cluster to which that sector belongs in a RAM (step 74). These information are then written in a sector to be written next in the cluster in which user data is being written to create a bad information sector with an attribute flag for "bad sector information" (step 75). Thereafter, the location of the bad information sector (cluster number and sector number) is registered in a bad sector information location table (step 76). The bad sector information location table records the relationship between the cluster number in the bad information sector and the location of that sector. For example, registered in an entry determined by the cluster number is the location of the bad information sector including that number.

As described, a bad information sector is created in the flash memory when a bad sector is found so that the bad sector information can be maintained even if power failure should occur in a period before the related bad sector map is updated.

4) The operation of the SSF will be explained by referring to Figures 8 - 11 in erasing and initializing a cluster (an erase program and an initialize program executed by the controller).

Figure 8 shows a flowchart for erasure of a cluster. When a cluster Cn satisfies a condition such that the number of its valid sectors is below a predetermined number, the controller determines that cluster to be erased (step 80). Then, it performs garbage collection for saving valid data in that cluster in another cluster, which is described later (step 81).

After the garbage collection, the controller reads the CIS of the cluster Cn and saves it in the RAM. Then, the cluster number of Cn and the management information just read are written into a sector to be

written next in the cluster in which the user data is being written, and a "cluster information copy" flag and a calculated ECC are written in it so as to create a cluster information copy sector (step 83). After the location of the created CIS copy sector is registered in the CIS-copy location table, the controller erases the cluster Cn (steps 84 and 85). The CIS-copy location table records the relationship between the cluster numbers included in the copy sector and the location of that sector. For example, registered in an entry determined by a cluster number is the location of a CIS copy sector including that number.

Thus, the CIS copy sector is created in the flash memory before erasing a cluster so that the cluster management information is not lost even if power failure should occur during erasure.

Figure 9 shows a flowchart for initialization of a cluster. The initialization herein means that a cluster information sector is created in an erased cluster so that it is put into a state allowing the writing of user data. When a condition is reached such that clusters consisting of blank sectors only are exhausted, the controller selects a cluster Cm to be initialized from one or more erased clusters by taking the erase count or the like into consideration (step 90). Then, it reads the CIS copy sector on the cluster Cm indicated in the CIS-copy location table to reconstruct the management information of the cluster Cm. The erase count is incremented at this time (step 91). Then, it reads the bad information sectors in the cluster Cm indicated in the bad sector information location table, and updates the bad sector map based on the information on the bad information sectors (step 92). Thereafter, a cluster information sector is created in the top sector of the cluster Cm. If writing in the sector should fail, or if the management information in the CIS copy sector (for example, the top bit of the bad sector map) indicates that the top sector is bad, the cluster information sector is created at the second location or later which is not bad (step 93). The location information on the CIS copy sector reflected in the cluster information sector is cleared from the CIS-copy location table, and the location information on the bad sector information location table is cleared from the bad sector information location table (step 94).

Now, the garbage collection is described in detail for two cases. Figure 10 shows a flowchart for a case where a data area is read from top to end. If it is determined to be a CIS copy sector from the attribute flag of the read sector, the controller determines whether or not it holds valid information, and therefore, has to be copied to another sector (step 102). More specifically, the erase count Ex of a cluster Cx held by the CIS copy sector is compared with an erase count Ea actually written in the CIS of cluster Cx. If Ex equals Ea, the CIS copy sector is considered to be valid data. Also, if the cluster Cx is in an erased state, it is considered to be valid. Otherwise, it is handled as

invalid data. Thus, it prevents the CIS copy sector from being erased as invalid data. If the cluster is in an erased state, judgment is made in step 142 in Figure 14, which will be described later.

Also, if the read sector is found to be a bad information sector, a similar judgment is made (step 104). More specifically, the erase count held by the bad information sector is compared with that written in the actual CIS. If both are equal, the bad sector information is considered to be valid. Otherwise, it is determined that the bad sector information has been reflected in the CIS, and is treated as invalid data. Thus, it prevents the bad information sector from being erased prior to update of the bad sector map.

Also, if the read sector is a data sector, it is determined whether or not it should be copied (step 105). More specifically, the address translation table is referenced by the reverse pointer to find a physical address written in it, which is then compared with its physical address. If both match, then it is valid data. Thus, the sector is copied in a location where the user data should be written in the cluster in which the user data is being written (step 106). Then, if the read sector is a valid CIS copy sector, the CIS-copy location table is updated, and if it is a valid bad information sector, the bad sector information location table is updated (step 107).

5) Figure 11 shows a flowchart for garbage collection which, when the validity of each data sector has been previously recorded, is performed by utilizing such record, the CIS-copy location table and the bad sector information location table. The validity is determined in a manner similar to step 102 in Figure 10 if the CIS copy sector is in a cluster to be erased, and in a manner similar to step 104 if there is a bad information sector (step 112 and 117).

The operation of the SSF will be explained by referring to Figures 12 and 13 in reconstructing the address translation table (an address translation table reconstruction program executed by the controller). The address translation table is essential in executing the dynamic sector allocation, but, because it is created on the RAM, is lost when the power is turned off. Therefore, when starting the system, the address translation table is reconstructed by reading all sectors in all clusters.

After successful reading of the CIS at the top of a given cluster, the controller sequentially reads the sectors in a data area from top to end (steps, 121, 122, 132 and 135). For a sector failed in reading, its location is temporarily saved in the RAM (step 131). For a sector successful in reading, its attribute flag is checked. If it is a CIS copy sector or a bad information sector, its validity is determined as in steps 102 and 104 (steps 125 and 128). If the read sector is a valid CIS copy sector, the CIS-copy location table is updated, and, if it is a valid bad information sector, the bad sector information location table is updated (steps

126 and 129). If it is a data sector, its physical address is registered in an entry of the address translation table pointed by the reverse pointer (step 130).

After completion of reading of the data area, the location of the read failed sector saved in step 131 is compared with the location registered in the bad sector map in the CIS (step 133). If the location of the read failed sector is not yet registered, an error message is displayed on, for example, the display device 24 (Figure 1) (step 134).

6) As described earlier, the SSF reads the cluster information sector when erasing a cluster, or when reconstructing the address translation table. However, if erasure or initialization has ended in failure due to system failure or the like, the content of cluster management information is broken, and an ECC uncorrectable error is generated. That is, reading of the top sector of the cluster has ended in failure. Also, in this embodiment, when the top sector of the cluster is bad, the cluster information sector is the next sector, or the sector at the top of data area. In this case also, the reading of the top sector has ended in failure.

Figure 14 shows a flowchart of an operation for acquiring the cluster management information when reading the top sector, or when failing such reading in erasing a cluster or in reconstructing the address translation table. If no uncorrectable error is detected when reading the top sector, step 146 checks the attribute flag. If it is confirmed to be "cluster information," the management information is saved in the RAM by type (step 147).

Situations where an error should be detected in step 141 includes such cases where the cluster has been erased. Then, step 142 determined whether or not the cluster has been erased. More specifically, the bit pattern of the top sector is compared with a bit pattern specific for an erased one. If both match, it means erased, and the process terminates immediately.

In this embodiment, if the top sector is bad, a cluster information sector proceeds all other sectors with any attribute in the data area. Then, even when step 143 determines an error to be uncorrectable, the controller repeats reading of sectors and detection of an error until it finds a sector not causing an uncorrectable error (steps 144, 145, and 143).

If a cluster properly read as the result of decision in step 146 is not a cluster information sector, it means that the cluster information sector is not at a location where it should be originally placed. Such a situation is generated when system failure such as power off occurs during erasing a block including the cluster information sector or during initializing a cluster, or when the cluster information sector becomes unreadable. Then, in step 148, a cluster information copy sector saving the cluster management information is searched by scanning all sectors in all clusters. Once it is found, erasure is performed again, and initializa-

tion is performed by fetching the management information from the cluster information copy sector, copying it to the top sector of the cluster, and creating a cluster information sector. If writing in the top sector should end in failure, the next sector is made the cluster information sector (step 150). If there is no CIS copy sector, it means that a fatal error is caused. Then, a message indicating the fact is informed to the user, and an instruction from the user is waited for (step 151).

Now, the management of sectors and clusters in the second embodiment will be explained.

7) As shown in Figure 15, a block, a physical erase unit, is divided into N pages. The size of one page is $256 + \beta$ bytes ($256 =$ one half of sector size, and β being a redundancy area of several bytes used by the system).

How to constitute a page does not generally depend on the physical requirements of a chip being used. For example, in a case of a chip in which access is made on a byte-by-byte basis, it is sufficient to simply handle $256 + \beta$ bytes as a logical block. It may also be possible to use a dedicated chip the physical page length (word line length) of which is $256 + \beta$ bytes. Figures 15 and 16 show a case where the latter chip is used.

8) An even number of blocks constitutes a set to form a cluster which is a logical erase unit. The controller performs erasure on a cluster-by-cluster basis. As shown in Figure 16, these blocks are divided into groups of each half, each of which is allocated to a different chip.

Allocation of blocks forming a cluster to a chip is generally performed in any desired method. Herein, block allocation as shown in Figure 16 is employed to simultaneously activate two chips for the purpose of improvement of data transfer rate. That is, one block is formed by four blocks every two of which are allocated to different chips. This allows it to double the bus width between the controller and the flash memory. If double bus width is further required, it is sufficient to divide the blocks forming a cluster into groups of $1/4$, and allocate each of them to four different chips. In this case, the length of one page is made $128 + \gamma$.

9) One physical sector is always assigned to a plurality of blocks (in this case, two). Also, an ECC parity is appended to an entire sector.

The length of a physical sector is $512 + \alpha$ bytes, and two pages are necessary to hold it (hereinafter, two pages for holding one physical sector being called a "page pair"). In this embodiment, it is arranged so that each page is in a different chip and therefore in a different block by allocating each pair in Pa (x) and Pb (x) as shown in Figure 16.

10) A cluster information sector is formed in the top sector in each cluster (Pa (0) and Pb (0) in Figure 16) in which, as in the first embodiment, management

information inherent to each cluster such as the cluster erase count and the bad sector map in the cluster is stored by appending parity codes. Figure 17 shows the format of the cluster information sector of this embodiment. It differs from that of the first embodiment in that the cluster management information is written in each page of the page pair in duplicate.

An ECC parity is calculated for the entire duplicated management information in the page pair (including the "cluster information" attribute flag), and written in the redundancy area in the page pair ($\alpha = 2\beta$ bytes). One half of the bits in the calculated ECC parity are stored in Pa (0), and the other in Pb (0). Therefore, the ECC portions in Pa (0) and Pb (0) are different from each other. Except for the ECC portion, both pages hold exactly same content.

Because, as described, the cluster management information is held in two pages in duplicate, it is never lost all at once due to a word line failure. In addition, because these two pages are allocated to different blocks, the cluster management information is never lost all at once due to a block failure. In Figure 16, page pairs for sectors other than the cluster information sectors are allocated to two blocks. Generally, endurance against block failure can be sufficiently enhanced even if the page pair constituting the cluster information sector is allocated in different blocks, while page pairs for other sectors are allocated in a same block.

While, in the first embodiment, a sector other than the cluster information sector is never positioned at the top of the cluster, a replacement-CIS pointer sector may be positioned in it in this embodiment. As shown in Figure 18, the replacement-CIS pointer sector is such a one, if one of pages in the top sector of the cluster is bad, to hold the address of replacement-CIS in the other page. Figure 19 shows a format for it. The address (pointer) of a replacement sector for storing the cluster management information is written in the top of one page, and an attribute indicating it as the replacement sector pointer and an ECC are written in the redundancy area of β bytes.

Usually, the second sector in a cluster is selected as the replacement sector. However, if such sector has a bad sector map set with a "bad" flag, selection proceeds one after another for the replacement-CIS sector. A sector selected as the replacement sector has usually the same format as the ordinary cluster information sector. It has the same formats for the data sector, the cluster information copy sector, and the bad sector information sector as those shown in Figures 4 through 6.

11) In this embodiment, the operations of the SSF are same as those in the first embodiment when a bad data sector is found, or when erasure, initialization, or creation of the address translation table is performed. However, because the cluster information is duplicated and the replacement sector pointer is provided, the

operation differs from that of the first embodiment when the top sector of the cluster is read and when such reading is failed. Figures 20 and 21 show the flowchart of the operation in this embodiment.

When no uncorrectable error is detected by reading the top sector, the process proceeds to step 216 where the management information is divided and stored in the RAM by type.

If an uncorrectable error is detected, the cluster information sector is separately read by page. First, the page Pa (0) is read and checked for the integrity of data by the parity codes for Pa (0) appended to various management information (step 204). If they are correct data, the process proceeds to step 208 where the attribute flag of Pa (0) is checked.

If the data read from Pa (0) is incorrect, the page Pb (0) is read and checked by the parity codes (step 206). If the data is correct, the process proceeds to step 209 where the attribute flag of Pb (0) is checked.

If the contents of both page for the top sector are incorrect, a CIS copy sector is searched. If it is found, the process proceeds to step 212 where the CIS is recovered as in step 149 in Figure 14. If not, it proceeds to step 213 where processing such as error display is performed as in step 150 in Figure 15.

If the checking of attributes in the page Pa (0) or Pb (0) which is not bad in step 214 reveals that the page is one of the pages constituting the CIS, it means that neither a replacement-CIS sector or a replacement-CIS sector pointer has yet been created. Therefore, to create the replacement-CIS and the replacement-CIS pointer sector in subsequently initializing a cluster, which of pages for the number and the top sector of the cluster is bad is saved in the RAM (step 215). Then, in step 216, the management information contained in that page is divided and stored in the RAM by type.

If the attribute of Pa (0) or Pb (0) is a "replacement-CIS pointer," the process proceeds to step 217 where the replacement-CIS being pointed is read and the management information is fetched.

Although, in practice, there is a step corresponding to step 142 in Figure 14 between steps 201 and 202, for the convenience of description, it is omitted in Figure 20.

Although two specific embodiments are described in the above, the scope of this invention is not limited to them. For example, in an SSF in which a sequence number is written in a cluster and rewritten at every erasure or initialization as disclosed in Japanese Pat. Appln. No. 5-35228, it may be possible to compare the sequence number instead of the erase count in steps 112 and 117 in Figure 11 or in steps 125 and 128 in Figure 12.

A method for reliably attaining, saving, and updating control information which is essential to assuring the reliability of a solid state file apparatus (SSF) using nonvolatile memories with flash erase capability

has been described. A solid state file apparatus and means for recovering control information for erase counts of clusters or the like when it is lost by a system failure such as an accidental power failure has also been described. When a bad sector is detected during operation of the SSF, it is correctly reflected in the control information holding area. Means for holding and recovering the control information which can cope with a situation when the control information holding area becomes unusable due to a sector failure such as a word line failure was also described as was means for holding and recovering the control information which can cope with a block failure.

Thus a nonvolatile memory with flash erase capability described, comprises a plurality of sectors, each of said sectors holding the attribute information for identification. The nonvolatile memory with flash erase capability can be erased for each cluster as a unit, which comprises a plurality of sectors, each cluster being provided with a cluster information sector for storing its own control information, the cluster information sector for each cluster holding attribute information for identification, a sector to be written with user data holding attribute information for data sector. The cluster information sector is in principle placed at a predetermined location in a cluster to which it belongs. The data sector is placed in a data area in the cluster which is an area other than that predetermined location. A data area of a given cluster may include a copy of management information on other clusters. Written in this cluster are the identifier for the other clusters and attribute information indicating them as copies of the cluster information sector. The management information may include the erase counts of the cluster and a bad sector map. The data area of a given cluster may include a sector for storing a location of bad sector which is in that cluster or another cluster, and is not yet reflected in the bad sector map of a related cluster information sector. This sector is written with an identifier for the cluster including the bad sector and attribute information indicating it as a bad information sector.

The solid state file apparatus described, comprises a controller, and a nonvolatile memory with a flash erase capability connected to it, in each sector of which attribute information is set. The controller creates a cluster information copy sector in a data area upon cluster erasure, and reconstructs cluster management information from the cluster information copy sector upon cluster initialization to create a cluster information sector.

Claims

1. A nonvolatile memory having flash erase capability, comprising a plurality of sectors, each of the sectors storing attribute information for iden-

tification.

2. A nonvolatile memory as claimed in Claim 1 further comprising a plurality of clusters, each cluster comprising a plurality of said sectors, and the clusters are erasable on a cluster-by-cluster basis, and wherein each cluster is provided with a cluster information sector for storing management information for said cluster, said cluster information sector holding attribute information for the cluster information sector; and sectors for user data holding the attribute information for the data sector. 5
3. A nonvolatile memory as claimed in Claim 2, wherein said cluster information sectors are placed at a predetermined location in the clusters, said data sectors are placed in a data area at a location other than said predetermined location in the cluster. 10 15 20
4. A nonvolatile memory as claimed in Claim 3 further comprising a sector in a data area in a given cluster for storing a copy of management information for another cluster, said sector holding the identifier of said another cluster and the attribute information for the cluster information copy sector. 25
5. A nonvolatile memory as claimed in any of Claims 2 to 4 wherein said management information contains cluster erase counts and a bad sector map. 30
6. A nonvolatile memory as claimed in Claim 5 further comprising a sector in a data area in a given cluster for storing a location of a bad sector which is in said cluster or another cluster and not yet reflected in the bad sector map in the related cluster information sector, said sector holding the identifier of the cluster containing said bad sector and the attribute information for the bad sector information sector. 35 40
7. A nonvolatile memory as claimed in any of Claims 2 to 6 wherein each cluster comprises a plurality of blocks, each of said blocks being a physical erase unit and comprising a plurality of pages; 45
 - top pages of one half of the blocks constituting a given cluster being allocated to the first half of the cluster information sector of said given cluster, top pages of the remaining blocks being allocated to the second half of said cluster information sector; and 50
 - the management information for said given cluster being written in said first and second halves of said cluster information sector in dupli-

cate.

8. A nonvolatile memory as claimed in Claim 7, wherein the first and second halves of said cluster information sector are assigned to different memory chips.
9. A nonvolatile memory as claimed in Claim 8, wherein, when one half of said cluster information sector is bad, a replacement-cluster-information sector is placed in the data area of said given cluster, said sector holding the attribute information for cluster information sector; the other half of said cluster information sector holding a pointer pointing said replacement-cluster-information sector.
10. A solid state file apparatus comprising:
 - a controller; and
 - a nonvolatile memory with flash erase capability comprising a plurality of clusters, each of said clusters comprising a plurality of sectors, said clusters being erasable cluster-by-cluster basis, each cluster being provided with a cluster information sector for storing management information for said cluster, said cluster information sector holding the attribute information for cluster information sector, and sectors for user data holding the attribute information for data sector.
11. A solid state file apparatus as claimed in Claim 10, wherein said cluster information sectors are placed at a predetermined locations in the clusters, said data sectors being placed at data areas at locations other than said predetermined locations in the clusters.
12. A solid state file apparatus as claimed in Claim 10 or Claim 11, wherein said management information includes erase count of the cluster.
13. A solid state file apparatus as claimed in any of Claims 10 to 12, wherein:
 - prior to erasure of a given cluster, said controller reads the management information in its cluster information sector; and
 - writes an identifier of said given cluster and said management information in a sector in a data area of a cluster other than said given cluster, and sets in said written cluster the attribute information for cluster information copy selector.
14. A solid state file apparatus as claimed in Claim 13, wherein:
 - said management information includes cluster erase count;
 - said controller determines the validity of the cluster information sector in a given cluster to

be erased by comparing the erase count held in said cluster information copy sector with that held in the cluster information sector for the cluster pointed by said sector; and

if it is valid, copies said cluster information copy sector to a data area of another cluster prior to erasure of said given cluster.

15. A solid state file apparatus as claimed in Claim 13 or Claim 14, wherein, when initializing a given cluster after erasure, said controller reconstructs the management information from the cluster information copy sector, which is created prior to erasure and includes the identifier of said given cluster, and creates the cluster information sector in said given cluster.
16. A solid state file apparatus as claimed in any of Claims 13 to 15, wherein, when failing to read the cluster information sector for a given cluster, said controller searches the cluster information copy sector including the identifier of said given cluster.
17. A solid state file apparatus as claimed in any of Claims 10 to 16, wherein said management information includes a bad sector map.
18. A solid state file apparatus as claimed in Claim 17, wherein, when detecting a bad sector in a given sector, said controller writes the identifier of said cluster, the identifier of said bad sector, and the erase count of said cluster in a sector of a cluster to which user data is being written, and sets in said written cluster the attribute information for the bad sector information cluster.
19. A solid state file apparatus as claimed in Claim 18, wherein:
 - said management information includes cluster erase count;
 - said controller determines the validity of the bad sector information sector in a given cluster to be erased by comparing the erase count held in said bad sector information sector with that held in the cluster information sector for the cluster pointed by said sector; and
 - if it is valid, copies said bad sector information sector to a data area of another cluster prior to erasure of said given cluster.
20. A solid state file apparatus as claimed in Claim 18, wherein, when initializing a given cluster after erasure, said controller references the bad sector information sector, which is created prior to erasure and includes the identifier of said given cluster, and updates the bad sector map of said given cluster.

21. A solid state file apparatus as claimed in any of Claims 10 to 20, further comprising:

a random access memory connected to said controller;

said random access memory being provided with an address translation table, said table translating a logical address contained in a command issued by a processor connected to said controller into a physical address pointing a specific sector;

when reconstructing said address translation table by reading all sectors in said memory upon starting up, said controller recording the associativity between the cluster identifier included in the detected cluster information copy sector and the location of said sector in a table other than said address translation table on said random access memory.

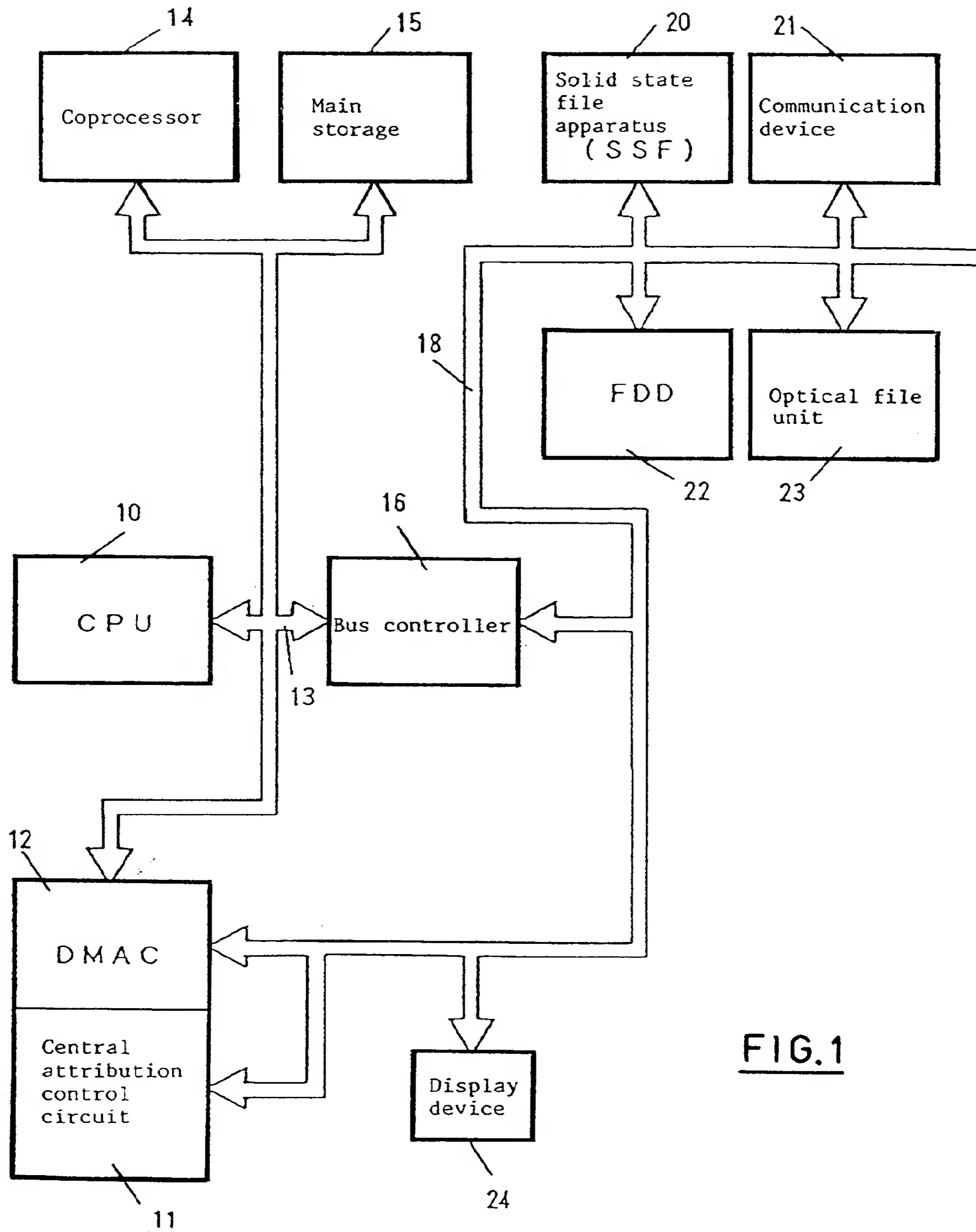


FIG.1

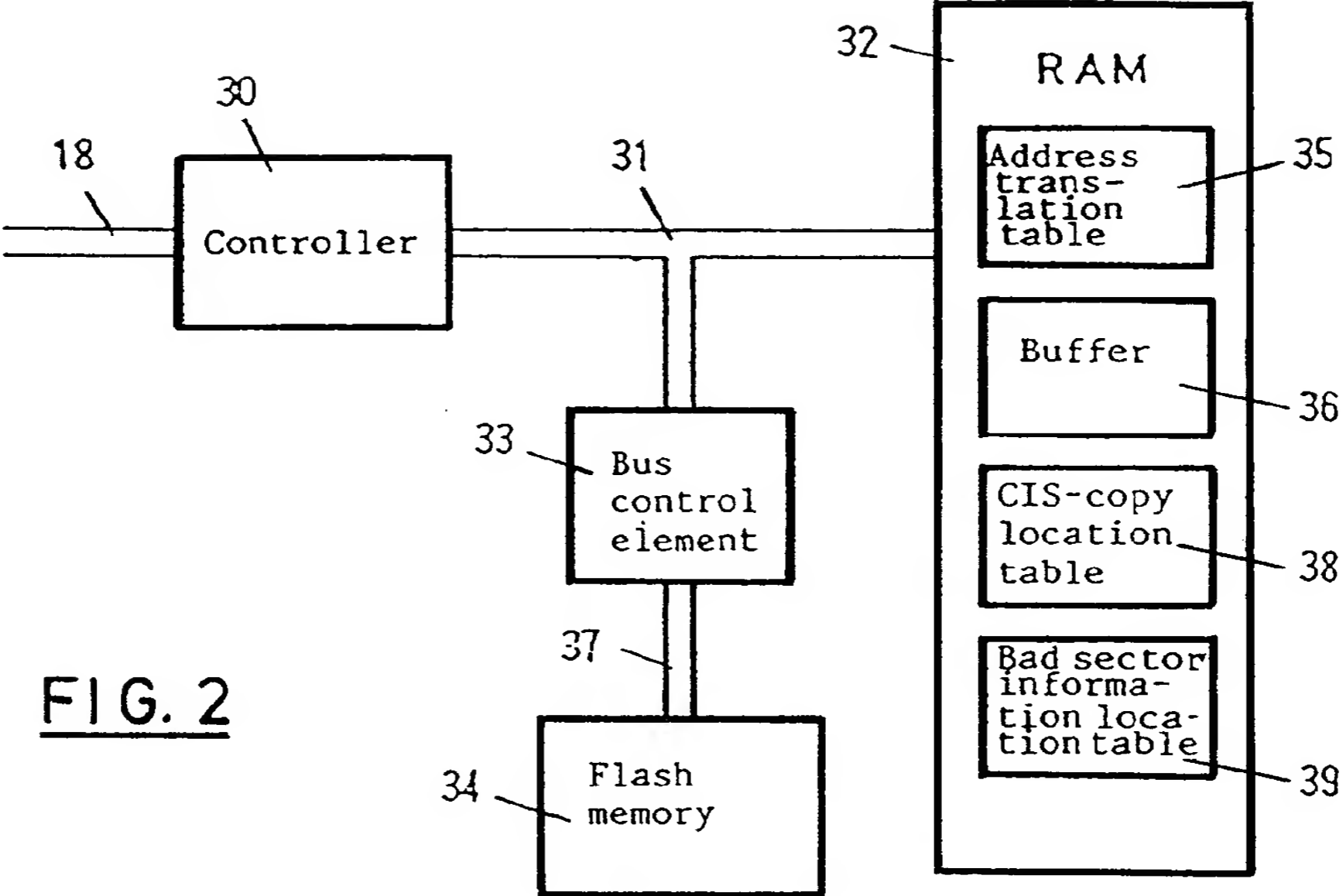


FIG. 3

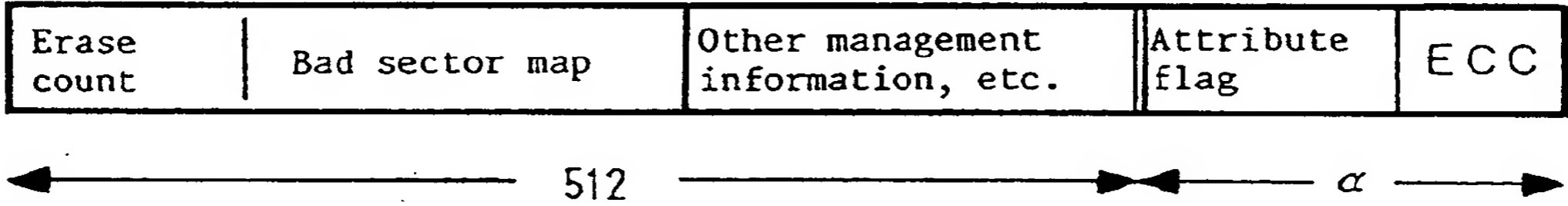


FIG. 4

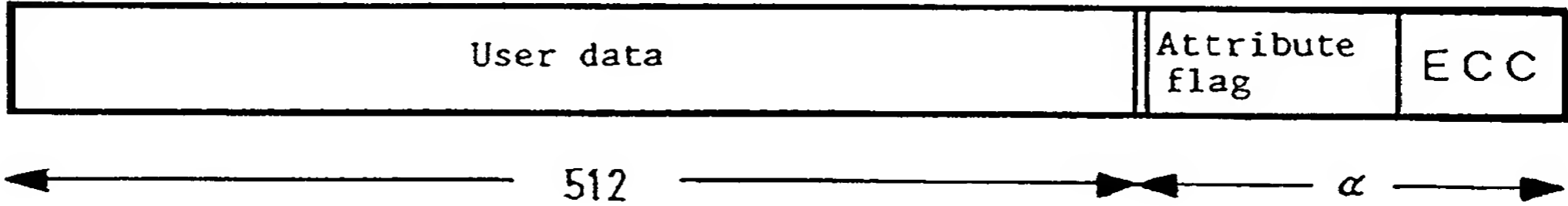
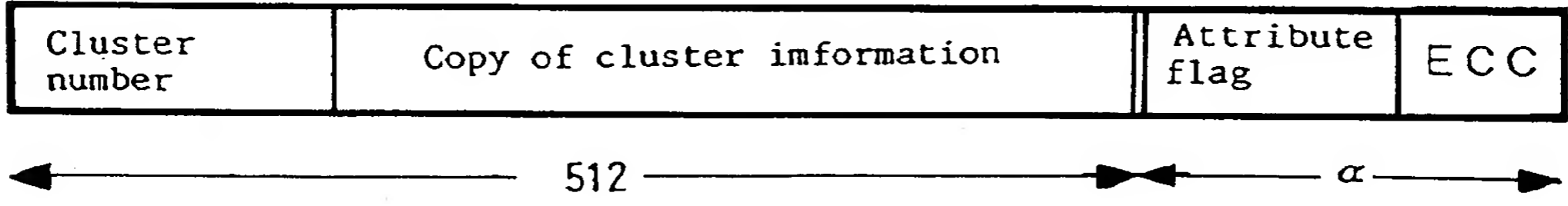


FIG. 5



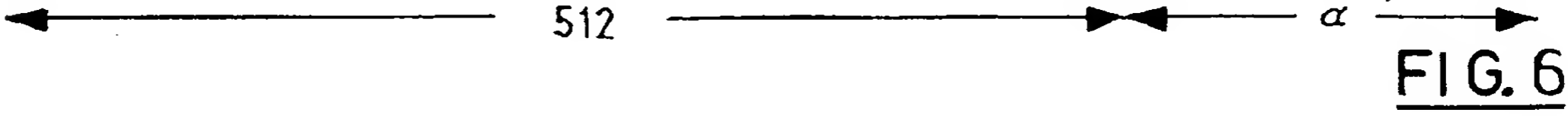


FIG. 6

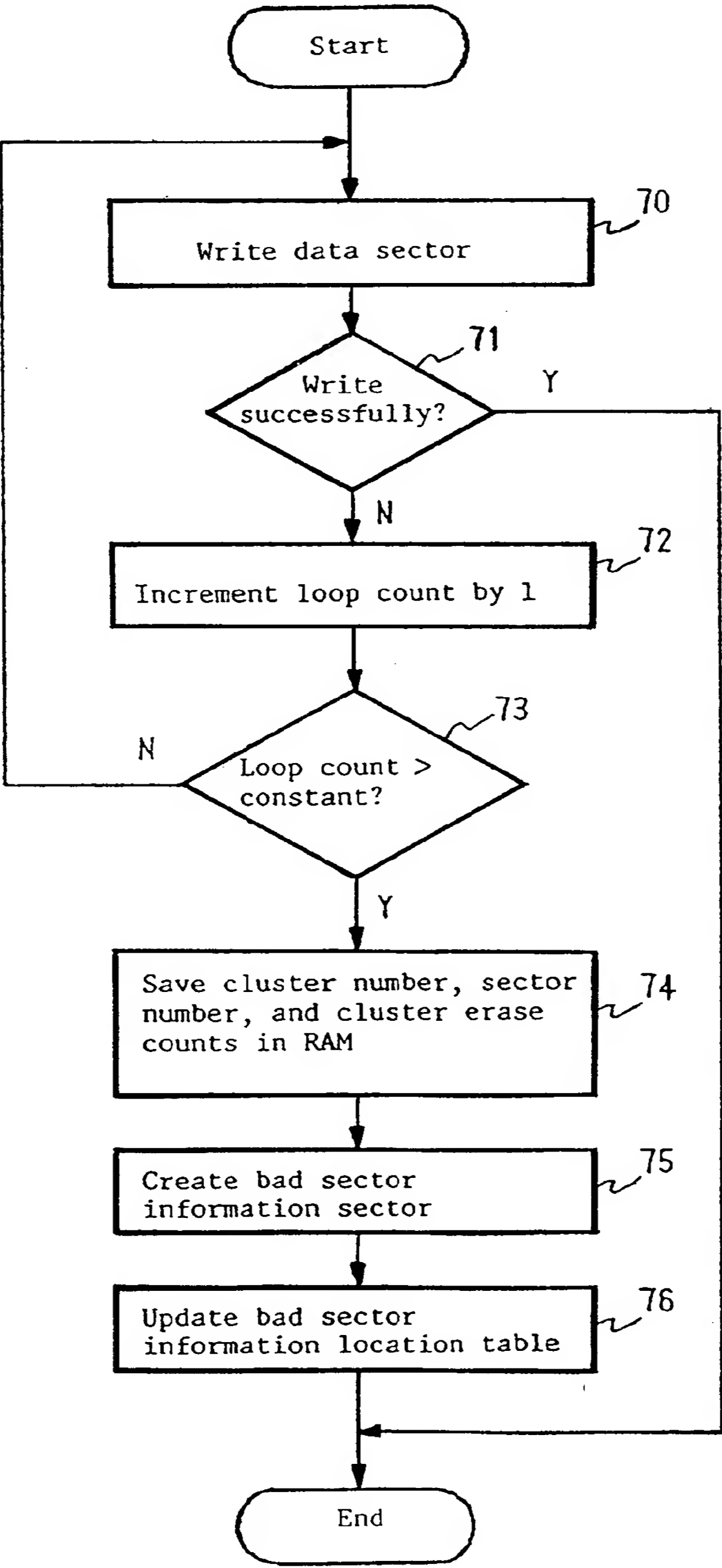
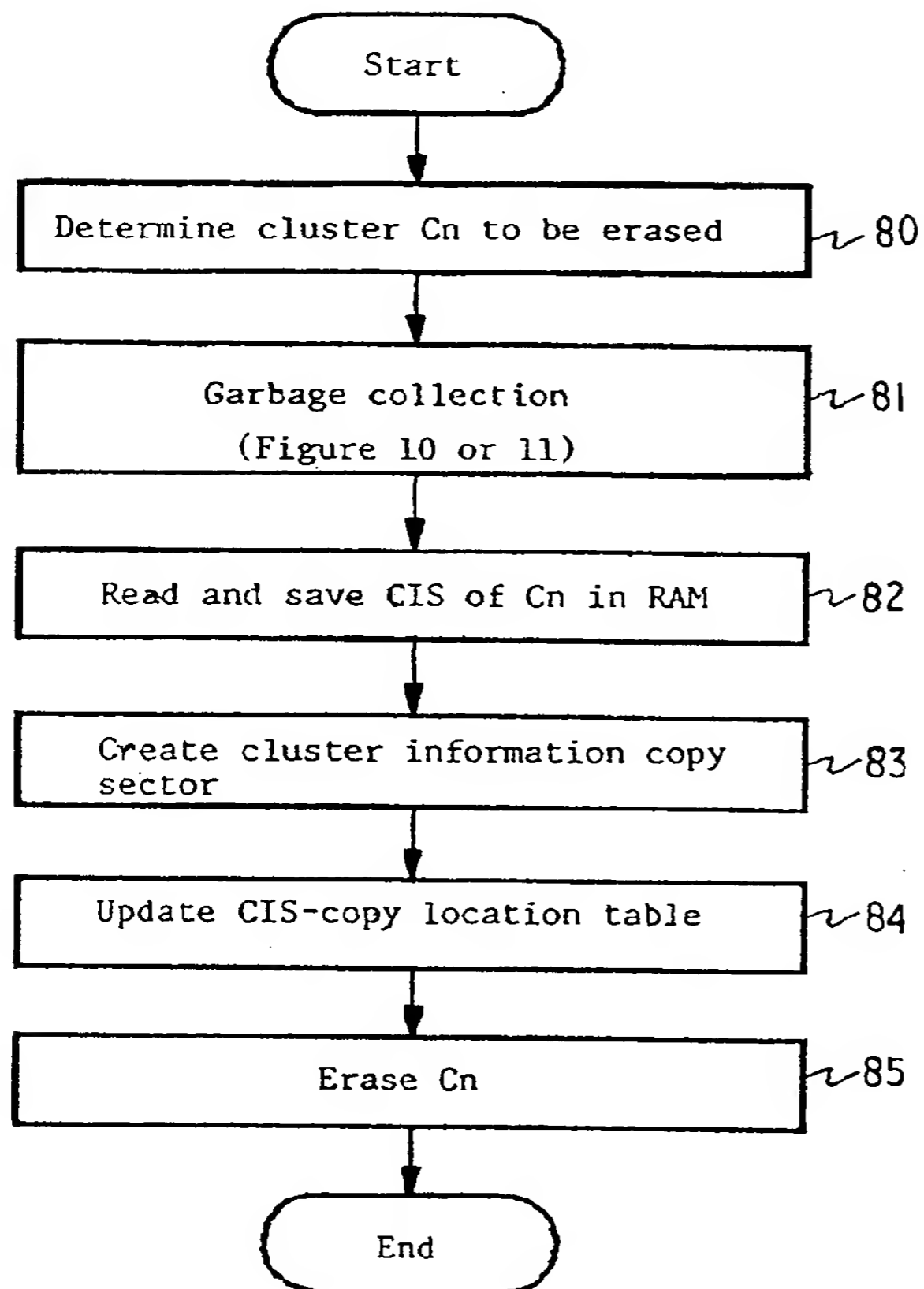
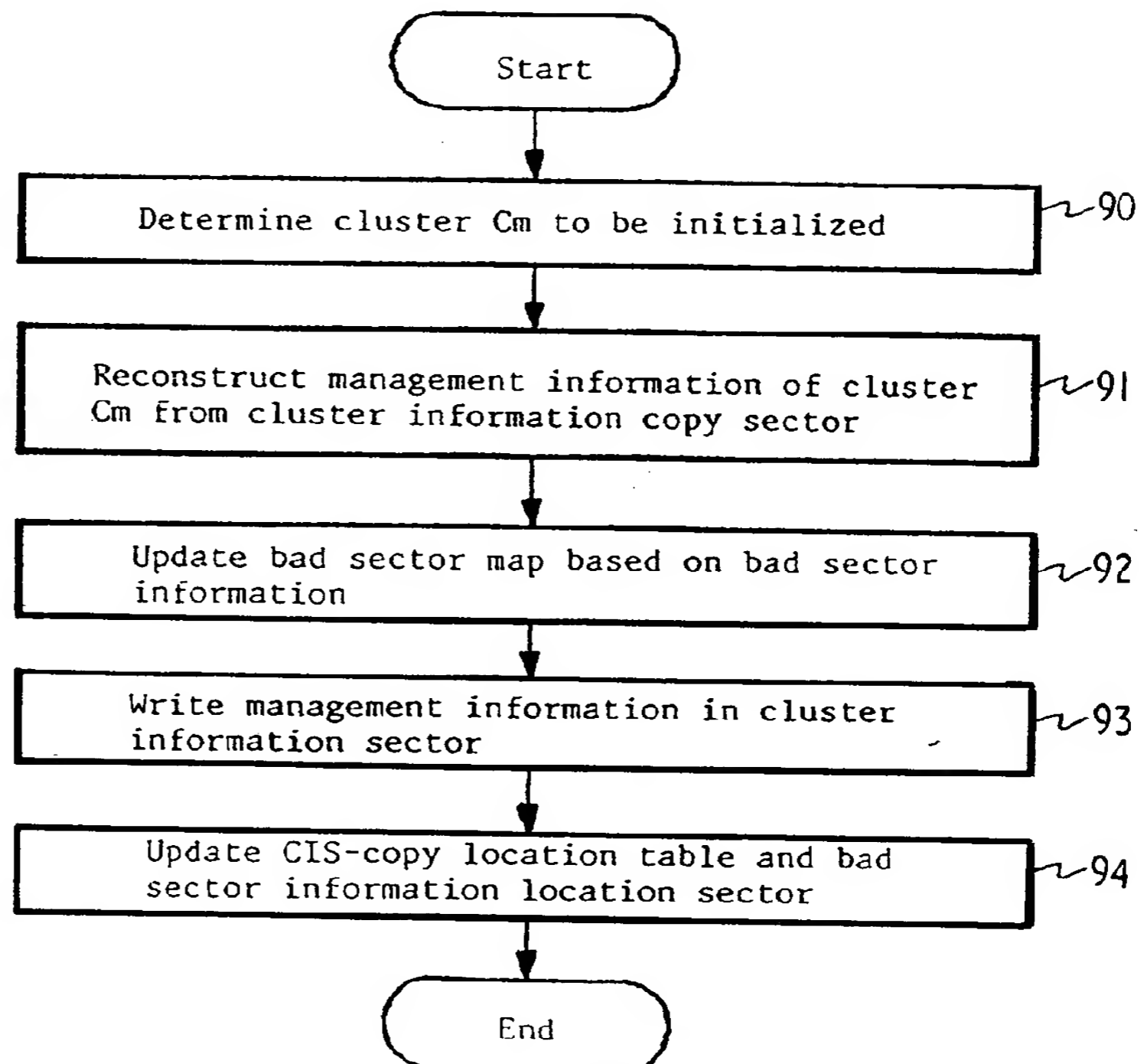


FIG. 7

FIG. 8FIG. 9

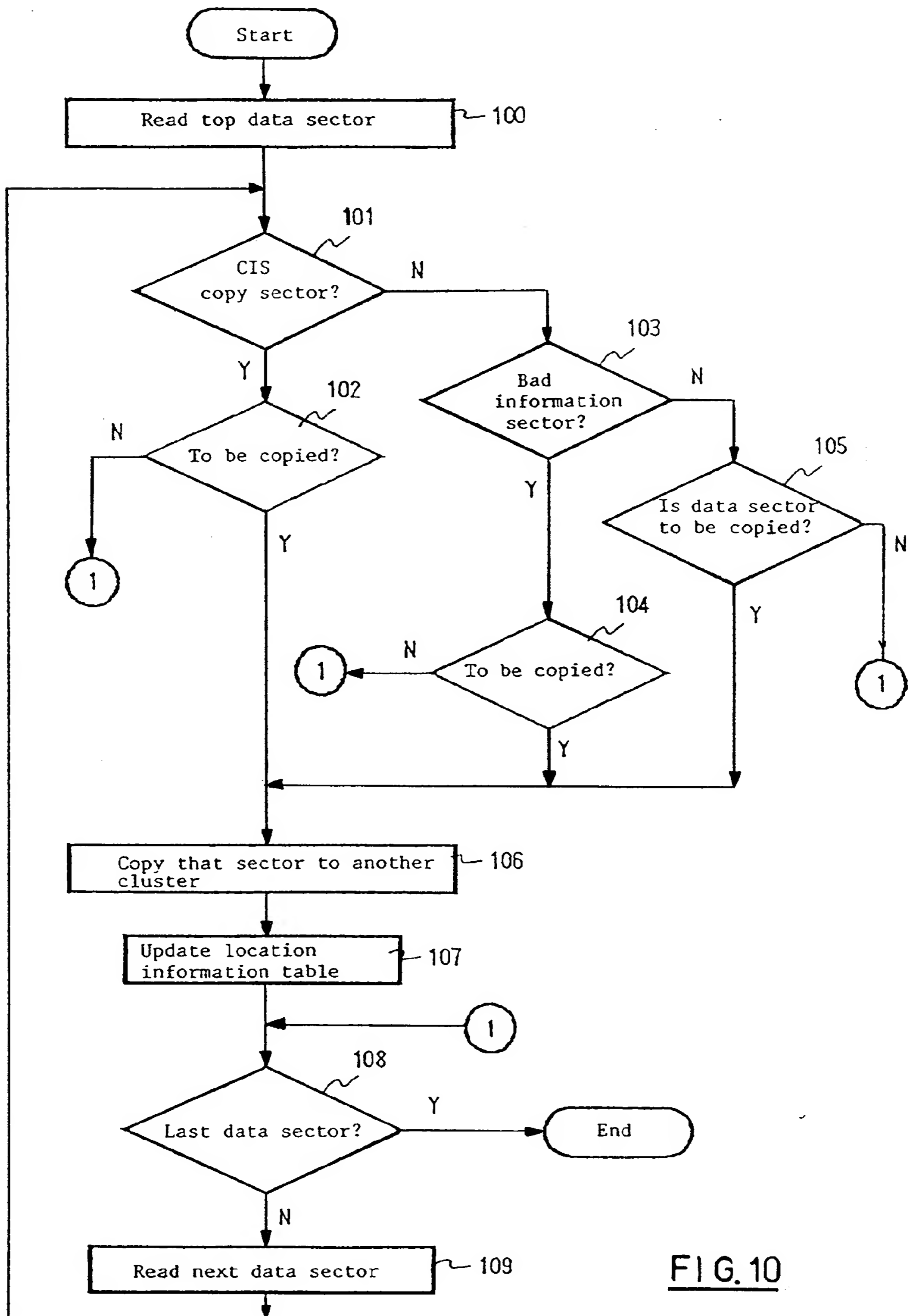


FIG. 10

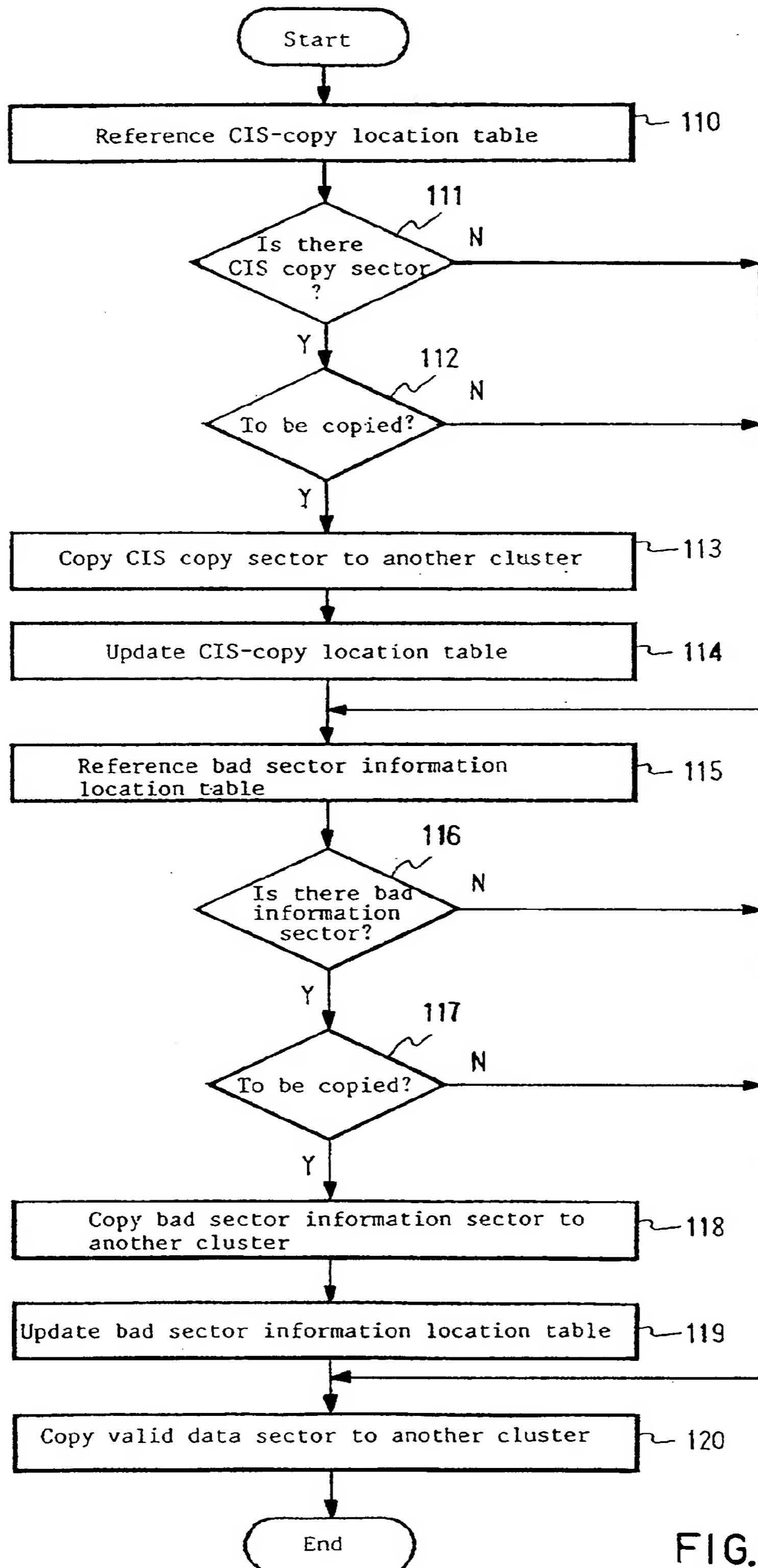
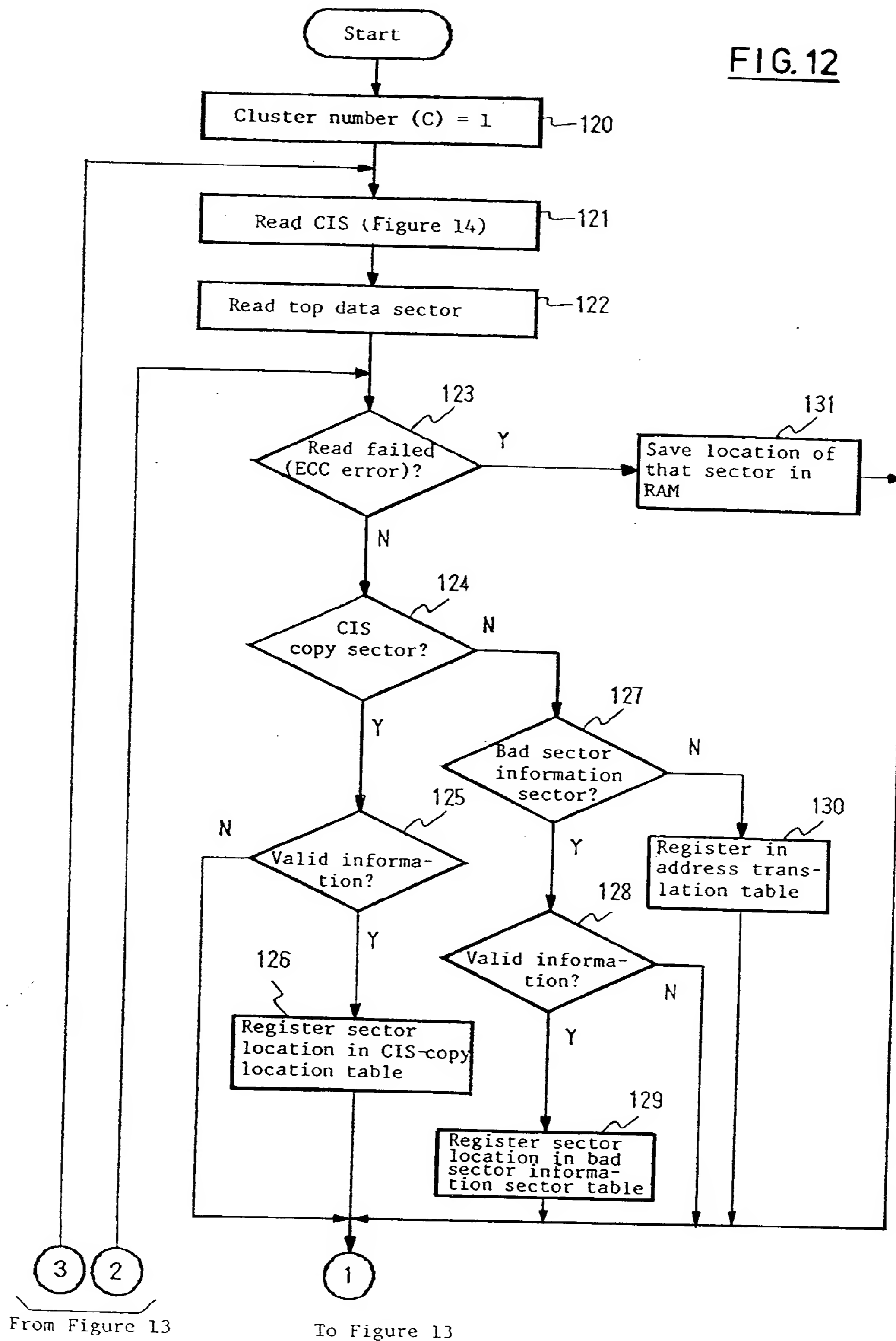


FIG. 11

FIG. 12



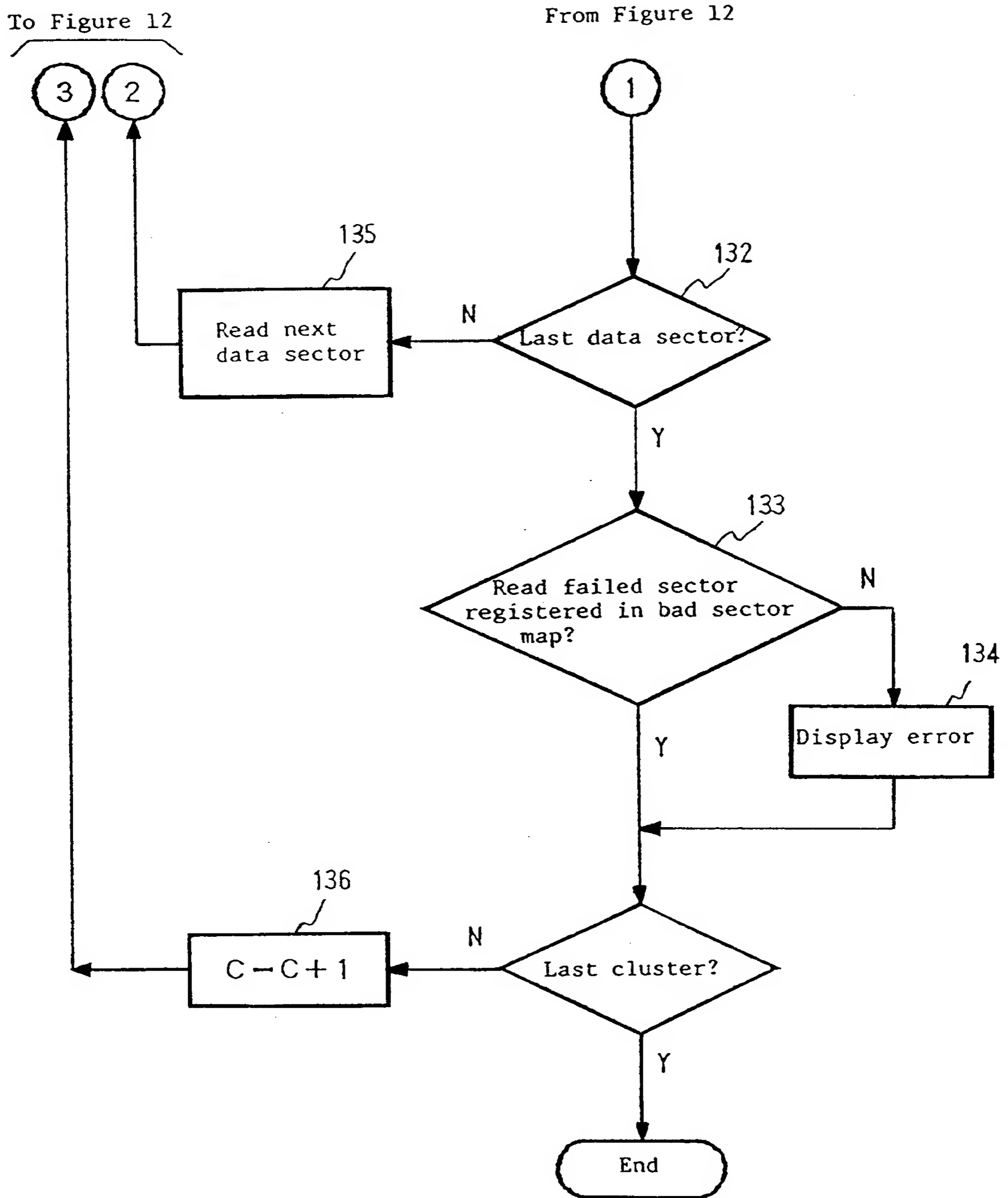
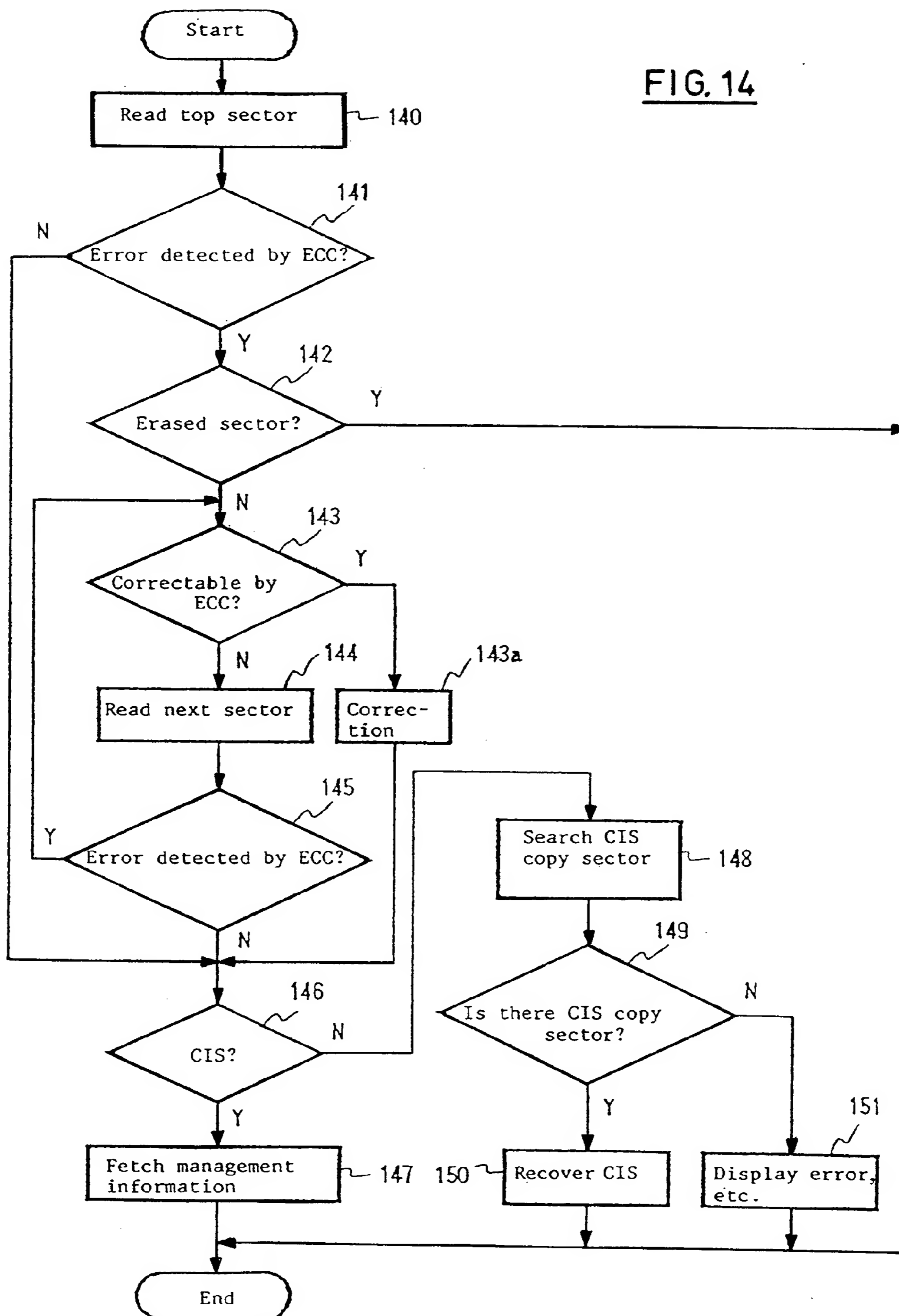


FIG. 13

FIG. 14



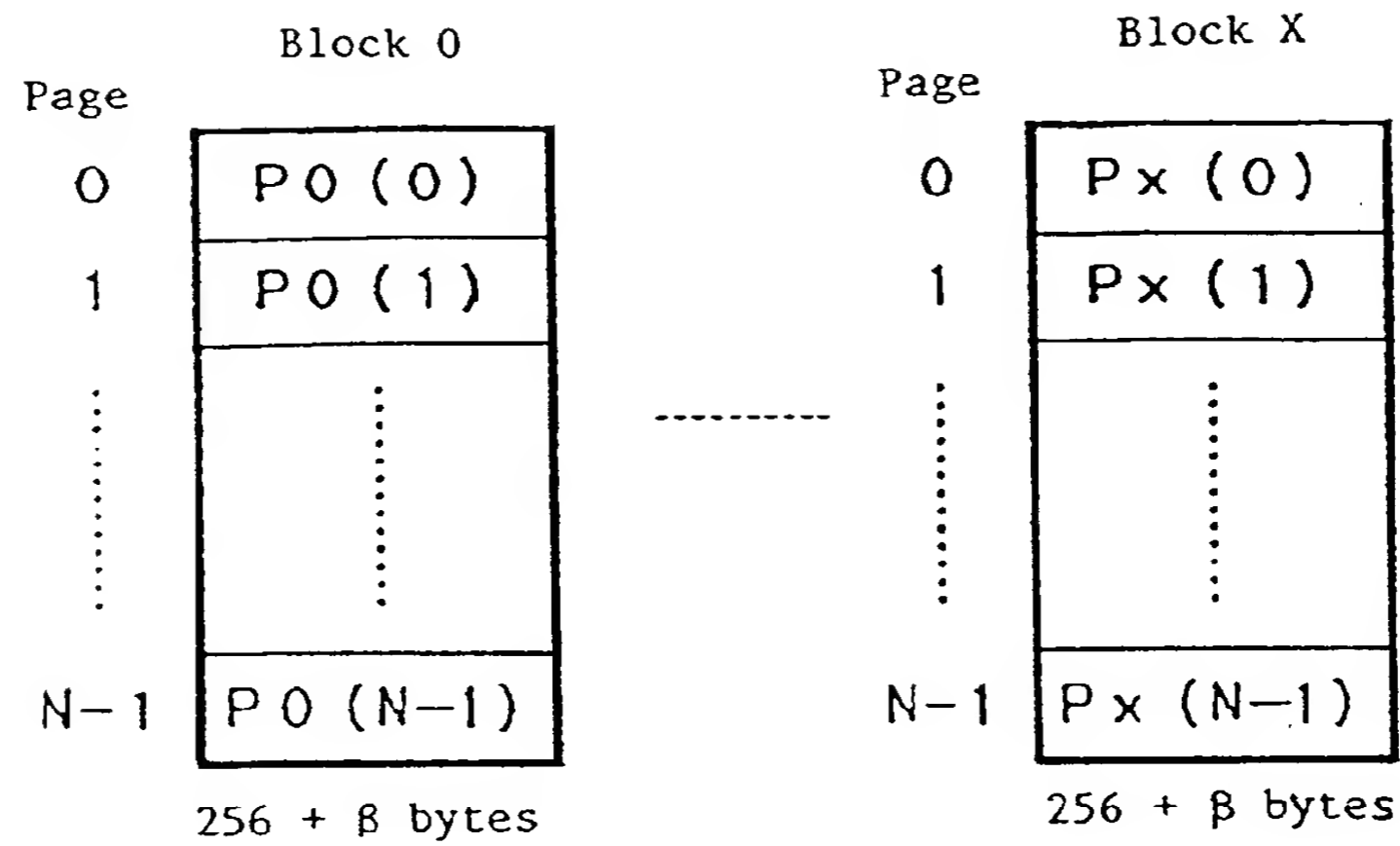


FIG. 15

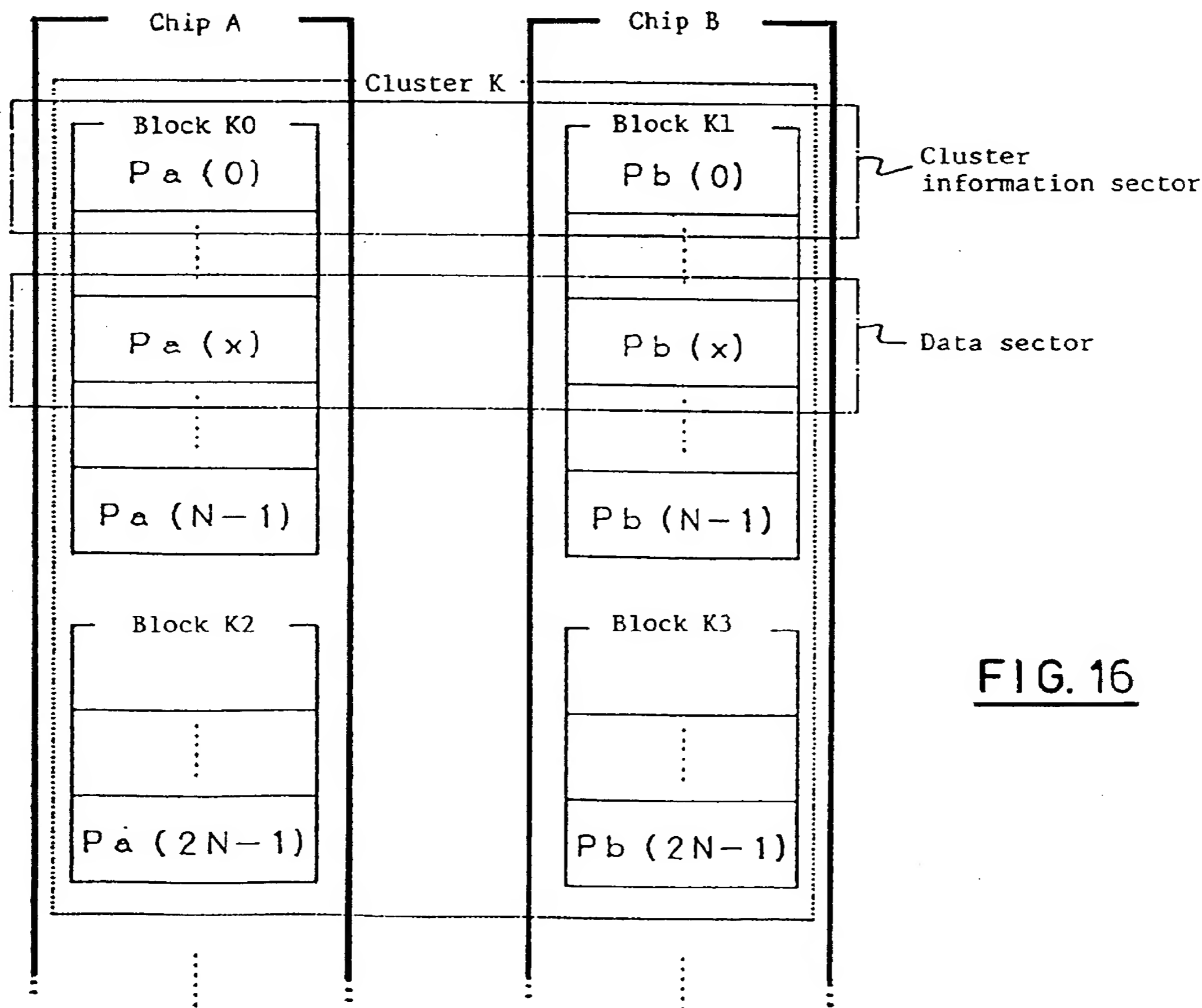


FIG. 16

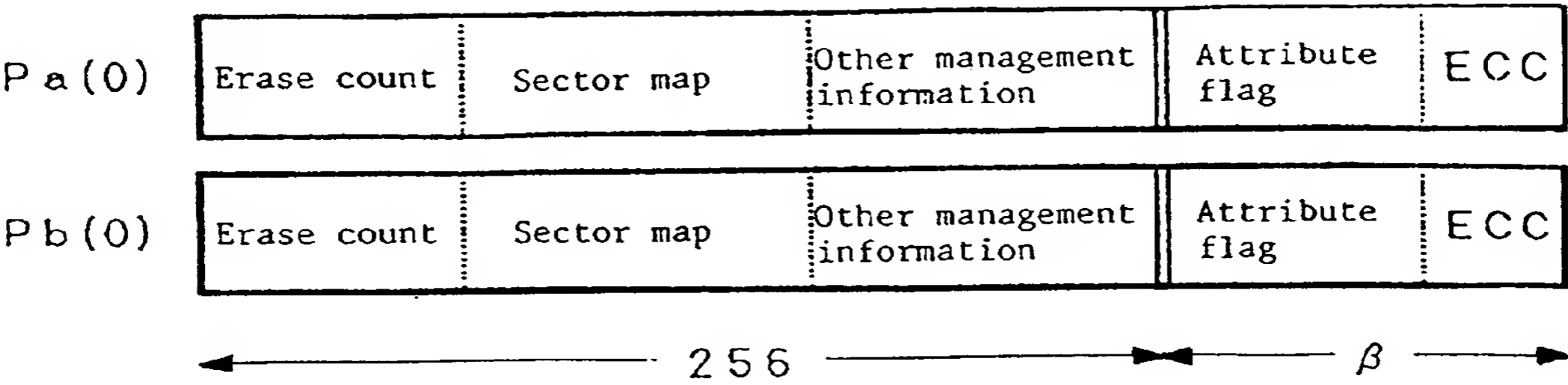


FIG. 17

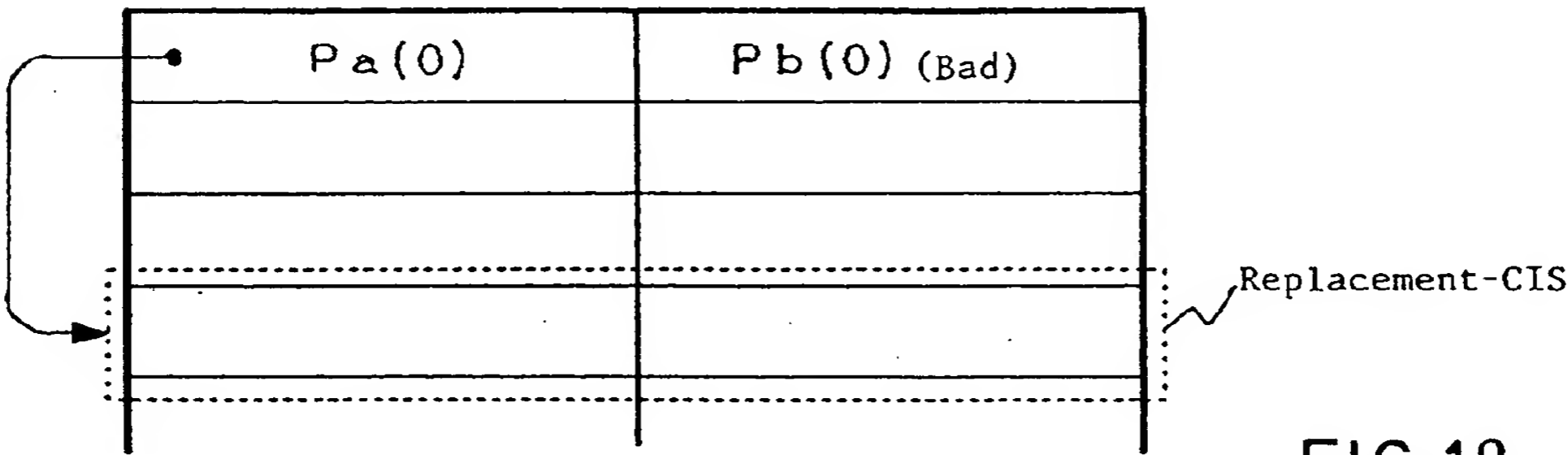


FIG. 18

FIG. 19

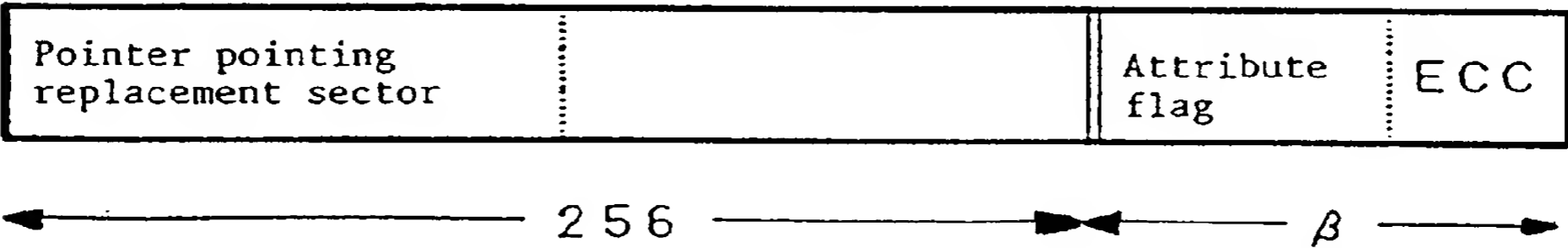
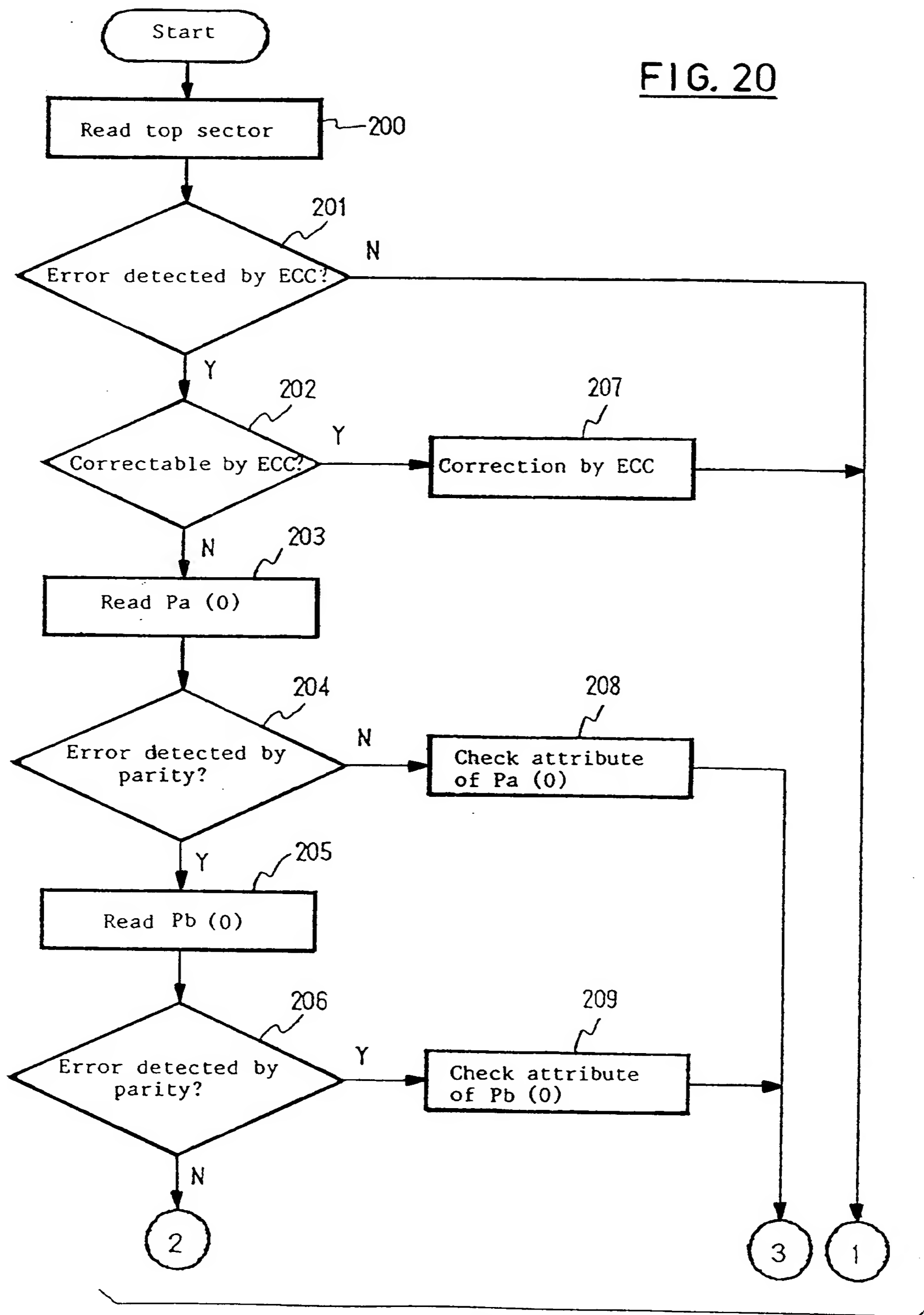
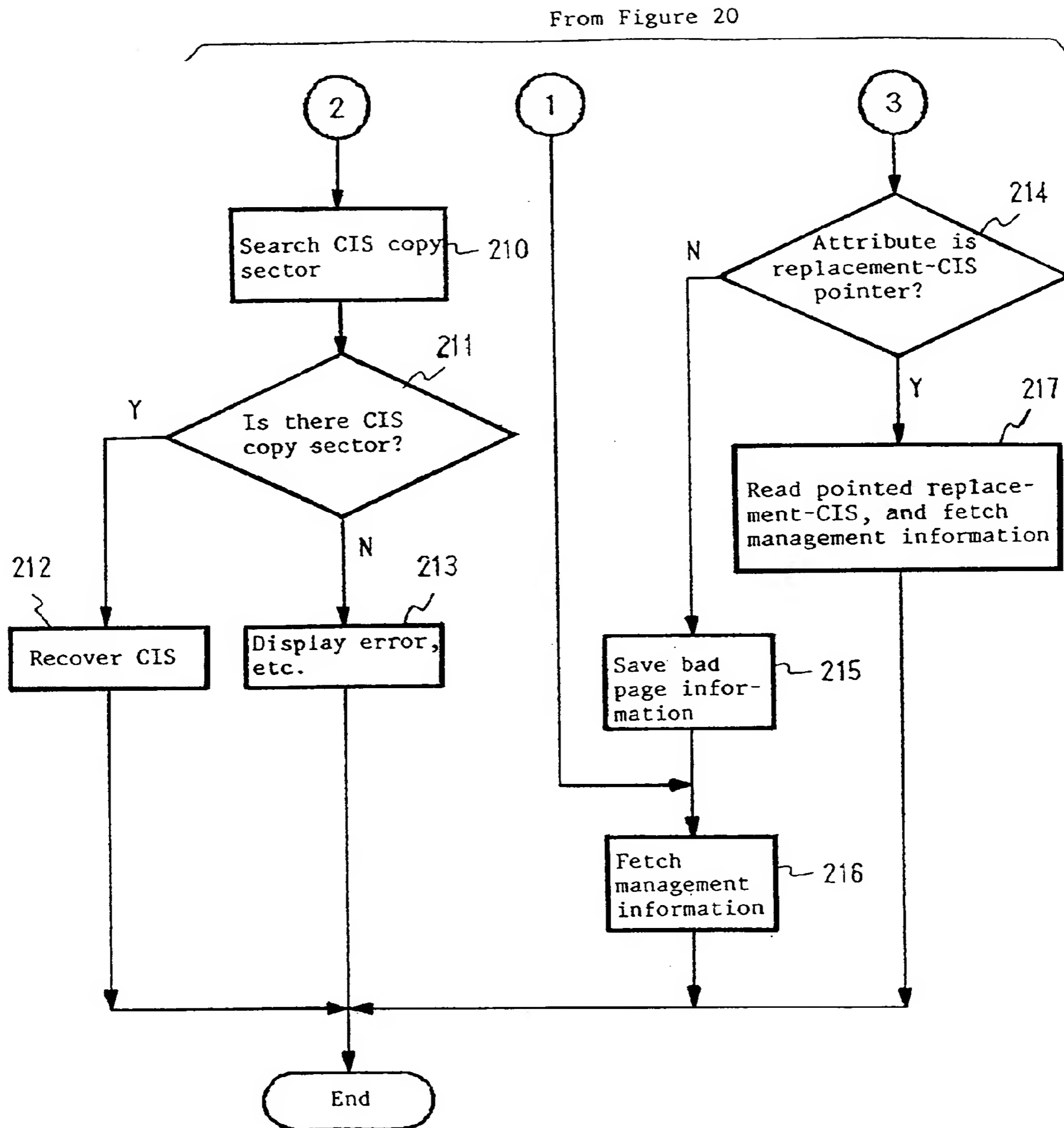


FIG. 20



**FIG. 21**